FERMILAB-Pub-03/325-E

Trigger Processor Algorithm for a Tracking Detector in a Solenoidal Magnetic Field

Marvin Johnson

Abstract-- **This paper describes a method of calculating triggers for a cylindrical multilayer tracking detector in a solenoidal magnetic field. The method is serial and requires one clock cycle per layer. It has two main advantages over a logical equation approach. First, missing track elements have no effect on the amount of calculation required and the resolution degrades in a predictable manner. Second, this method allows implementing information from the third (non-bend) coordinate as a simple extension to the basic algorithm.**

I. Introduction

The triggering of multilayer cylindrical tracking detectors used in high energy colliding beam experiments is based, to a large extent, on the prompt identification of charged tracks of a specific transverse momentum. This information is usually combined with information from other detector elements (e.g. energy deposition in calorimeters and/or presence of highly penetrating particles (muons)) to form a second level trigger decision that usually initiates the full readout of the detector(s).

The advent of large Field Programmable Gate Arrays (FPGAs) has created new opportunities for designing such trigger processors for this kind of tracking detectors. One method, used in the D0 experiment [1-3], is to create a list of logical equations of tracking detector elements that encompass all possible tracks for a given momentum range. For example, if one has a three layer detector, a logical equation would be: ((Layer 1 hit) AND (Layer 2 hit) AND (layer 3 hit)). The logical equation is chosen so that the 3 layer hits represent a track of interest. All possible tracks of interest are included in the set of equations and all equations are evaluated simultaneously in a set of FPGAs. This method is fast and efficient but if the detector is inefficient, the number of equations can become very large. Second, if there is information on the third (non bend) coordinate (e.g. from timing), it is difficult to include it in the logical equations.

The method described in this paper is similar to the logical equation approach in that it also starts with a complete set of logical equations. It avoids the problems mentioned above but it takes longer to compute the trigger. It is a serial operation which processes one layer of a detector at a time so for an 8 layer detector it could take up to 8 times as long as for the logical equation approach. Note that the use of pipelining may allow both methods to process data at the same rate but this approach will always have a longer delay.

The next section describes the mathematical basis for the algorithm. Section III describes the algorithm and gives an example from the D0 collider experiment.

II. Mathematical Description

A track orbit for a track of transverse momentum p_t, passing through the origin and in a uniform magnetic field is described by the following equation (in two dimensions and in polar coordinates):

$$Sin(\phi - \phi_0) = \frac{kBr}{p_t} \qquad (1)$$

where r and ϕ are the radius and angle at any point on the track, ϕ_0 is the tangent of the track at the origin, k is a constant (k= 0.3 (GeV/c)/Tesla m), B is the magnetic field, and p_t is the transverse momentum. Note that r is radial position of the detector element, not the radius of the track. For high p_t tracks and modest size detectors such that the right hand side of (1) is small, this can be approximated as :

Manuscript received October 22, 2003. This work was supported by the U.S. Department of Energy under Contract No. DE-AC02-76CH03000.
Marvin Johnson is with the Fermi National Accelerator Laboratory, Batavia, IL 60510, USA (telephone 630-840-3168, e-mail: mjohnson @fnal.gov).

$$\phi - \phi_0 = \frac{kBr}{p_t} \qquad (2)$$

An example of a two layer tracking detector is shown in Fig. 1. The shaded circles represent the elements of a tracking detector and the lines represent the orbits of the minimum and maximum transverse momentum particles that can pass through these detector elements. The minimum momentum track (for clarity we discuss here tracks of one polarity, the extension to both polarities is trivial) is the track that passes through the trailing edge of the detector element for the inner layer and the leading edge of the detector element for the outer layer. Similarly, the maximum momentum track is the one that passes through the leading edge of the detector element for the inner layer and the trailing edge of the detector element for the outer layer. This shows that for detectors with elements of finite width there is a range of p_t and ϕ_0 that will give the same hit pattern.

If we solve (2) for ϕ_0, we can plot ϕ_0 as a function of p_t for any given r, ϕ pair. Fig. 2(a) shows this for the two edges of the detector element of the inner layer of Fig. 1. The two curves are parallel and their separation is equal to the range of ϕ_0 angles at the origin for tracks of a particular p_t that can pass through this detector element. From equation (2) it is seen that this is just ϕ(leading edge) - ϕ(trailing edge) which is determined by the size of the detector element and its radial position. Fig. 2(b) shows the same function for the two edges of the detector element for the outer layer. Fig. 2(c) shows the superposition of these two plots.

The shaded area in Fig. 2(c) shows the allowed range of ϕ_0 as a function of p_t for valid tracks that pass through both detector elements of the inner and the outer tracking layers and the origin. Fig. 3 shows the effect of adding a third detector element midway between the inner and outer layer. Again, the shaded region shows the allowed range of ϕ_0 and p_t for valid tracks.

The shaded area in these plots is always bounded by the maximum of the value of ϕ_0 computed for the leading edge of all the detector elements and the minimum of ϕ_0 for the trailing edges of all the detector elements. We define the minmax function as the minimum ϕ_0 for the trailing edges minus the maximum ϕ_0 for the leading edges of a set of detector elements. A valid track will then have a positive value of this function. Note that this function is zero at both the beginning and end of the allowed p_t range.

This stratagem can be exploited to determine all possible logical equations for a given set of detector elements. The simplest (but most computing intensive) is to pick a rough track road (one detector element per layer), compute the p_t for all possible pairs of lead-trail and trail-lead edges of these detector elements and then use the minmax function to determine which combinations select real tracks. The range of allowed p_t can also be determined. By systematically going through all detector elements in all layers, one can find every allowed road.

The minmax function can also give the relative contribution of a logical equation to the total trigger rate. For a given logical equation, all successful tracks will fall in the shaded region of a figure similar to Fig. 3. The contribution of any equation to the total trigger rate is then the shaded area of one logical equation to the sum of all the areas for all logical equations. The Monte Carlo calculates this by the number of successes for a given equation to the number of generated tracks. We can calculate this analytically by numerically integrating the minmax equation over the allowed range of p_t for each equation and normalize each equation to the sum of the integrals from all the equations. This allows us to delete equations that make a very small contribution to the total trigger rate.

The minmax function gives the contribution of a logical equation at a given value of p_t. By plotting the sum of all minmax functions with a value greater than 0 as a function of p_t one gets the trigger acceptance as a function of p_t (often called the trigger turn on curve). One can make the same plot for any equations that have been deleted from the equation set and see the impact of the deletions. The difference between the two curves is the new turn on curve. Again, since these are analytic calculations, they will give the same result as a Monte Carlo simulation.

If one knew the p_t of a track, one could determine if a set of hits formed a track simply by evaluating the minmax function. Of course, the p_t is in general not known. However, the shaded area for most logical equations spans a rather large range in p_t. If one sorts all the equations for a range of p_t (10 to 1000 GeV/c for example) according to starting and stopping p_t, one finds that large numbers of equations share a common momentum. Thus, it may be practical to have several simultaneous computation engines that evaluate the minmax function for a few selected values of p_t. Since all of these calculations are done for a specific p_t, all the values can be precomputed so that no actual calculations are required. The entire algorithm can then be reduced to memory look-ups.

are dual ported, the minimum table can be stored in the unused part of the maximum memory. All that is required is that the layer and fiber address be complemented before being sent to the memory. Thus, only one memory is required.

Next, even though that table I has 3 states (fiber 1, fiber 2 or nothing), only 2 are required. The register is initialized to 0 at the start and only valid hits will load new values. Thus, the maximum table can be indexed by fiber 1 (if both fibers are hit, we take the leading edge of fiber 1) and the minimum is indexed by fiber 2 (if both are hit, we take the trailing edge of fiber 2). If neither are hit, the layer is skipped. This reduces the memory by one additional bit to 256 by 4. Finally, the values in the table are often the same. That is, the maximum value for layer 7 may be the same for every column. It is straightforward to remove this redundant information. For this example, we were able to remove more than half the columns so that the 4 bit address could be reduced to 3. This left us with a single 128 by 3 bit memory. Of course, other detectors may not compress this well.

This algorithm makes heavy use of embedded memory in the FPGA while the logical equation approach uses no memory. The logical equations must be known at compile time so the FPGA compiler can be used to minimize the amount of resources needed. This approach uses large amounts of memory which is not loaded until the part is ready for use so the compiler cannot optimize it. Because of this optimizing feature, the logical equation approach may use less resources than this method.

There are typically a very large number of equations for each inner–outer layer pair. However, many of them have a common momentum and those that fall outside of these large groupings often make negligible contributions to the total event acceptance. The latter can be dropped with little change in efficiency. For example, one 4.5 degree wedge of the D0 fiber tracker contains 8 layers with the inner layer having 16 fibers and the outer layer having 44 fibers. For tracks greater than 10 GeV/c, there are 233 inner-outer fiber pairs (2684 logical equations). If we restrict the loss for any given inner outer road to less than 2% (calculated by numerically integrating the minmax function), we get 826 roads. The average acceptance loss is less than 1%. To implement this method would then require 826 blocks similar to that shown in Fig. 5 plus the routing to get the proper fiber to the computing block and the logic necessary to gather the results. Note that this 2% loss is identical to losing 2% of the tracks in a Monte Carlo simulation. As mentioned above, one can see the affect of this loss on the trigger turn on curve by plotting the sum of the minmax function for all equations and the accepted ones as a function of p_t.

The transverse beam size (1 sigma) is around 33 microns for the Tevatron. If we displace the interaction point along the center of a sector by 2 sigma in all directions, we get 56 additional logical equations (2%). This is dependent both on the beam spot size and the resolution of the detector.

The block diagram in Fig. 5 does not use any z coordinate information. This is straight forward to add but it may require considerably more FPGA resources.

IV. Summary.

This method is an alternative to the logical equation approach and is suitable for trigger systems that have information from a third axis. The amount of logic required does not depend on the number of layers with hits so it may also be the method of choice for inefficient detectors. Since both methods depend in detail on the precise nature of the detector, one cannot recommend one over the other.

I would like to thank Petros Rapidis for many useful discussion and suggestions for this work.

V. References.

[1] J. Olsen *et al.*, "The D0 Central Track Trigger," *IEEE Trans. Nucl. Sci.*, vol. 51, pp. 345-350, June 2004.

[2] Borcherding, F.; Grunendahl, S.; Johnson, M.; Martin, M.; Olsen, J., Yip, K ., "A first level tracking trigger for the upgraded DØ detector, " *IEEE Trans. Nucl. Sci.*, vol. 46, pp. 359 –364, June 1999.

[3] Abbott, B.; Angstadt, B.; Borcherding, F.; Johnson, M.; Martin, M., "A fast, first level, rϕ hardware trigger for the D0 central fiber tracker using field programmable gate arrays," *IEEE Trans. Nucl. Sci.*, vol. 44, pp. 354 –357, June 1997.

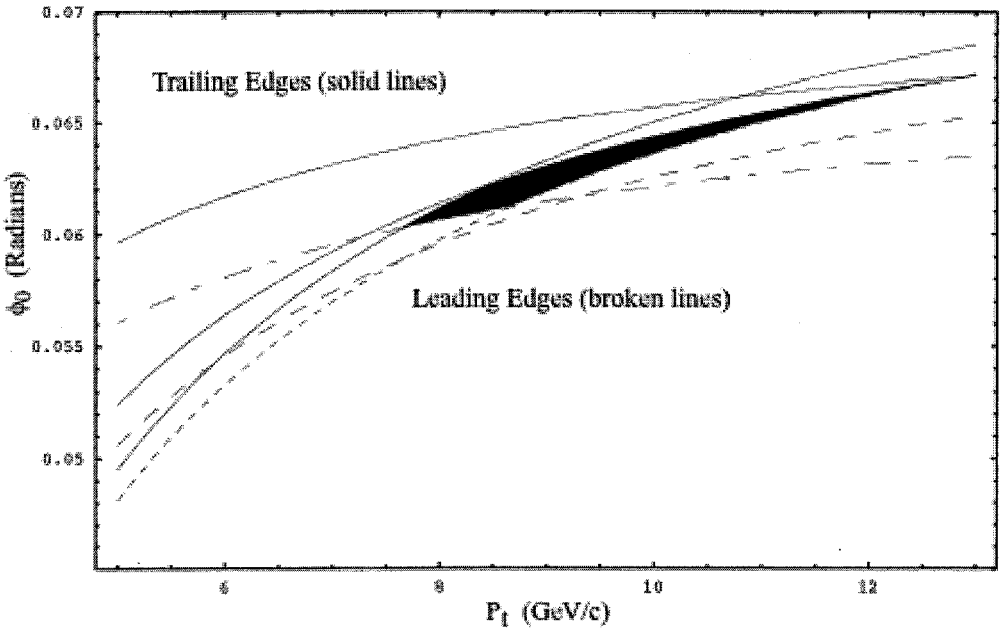

Fig. 3. This figure shows the effect of adding a third tracking layer to the data of Fig. 2(c). This detector element is positioned midway between the inner and outer elements shown in Fig. 1 (for the parameters used in this illustration see the caption for Fig. 1). A valid track can only exist if it lies in the shaded area.

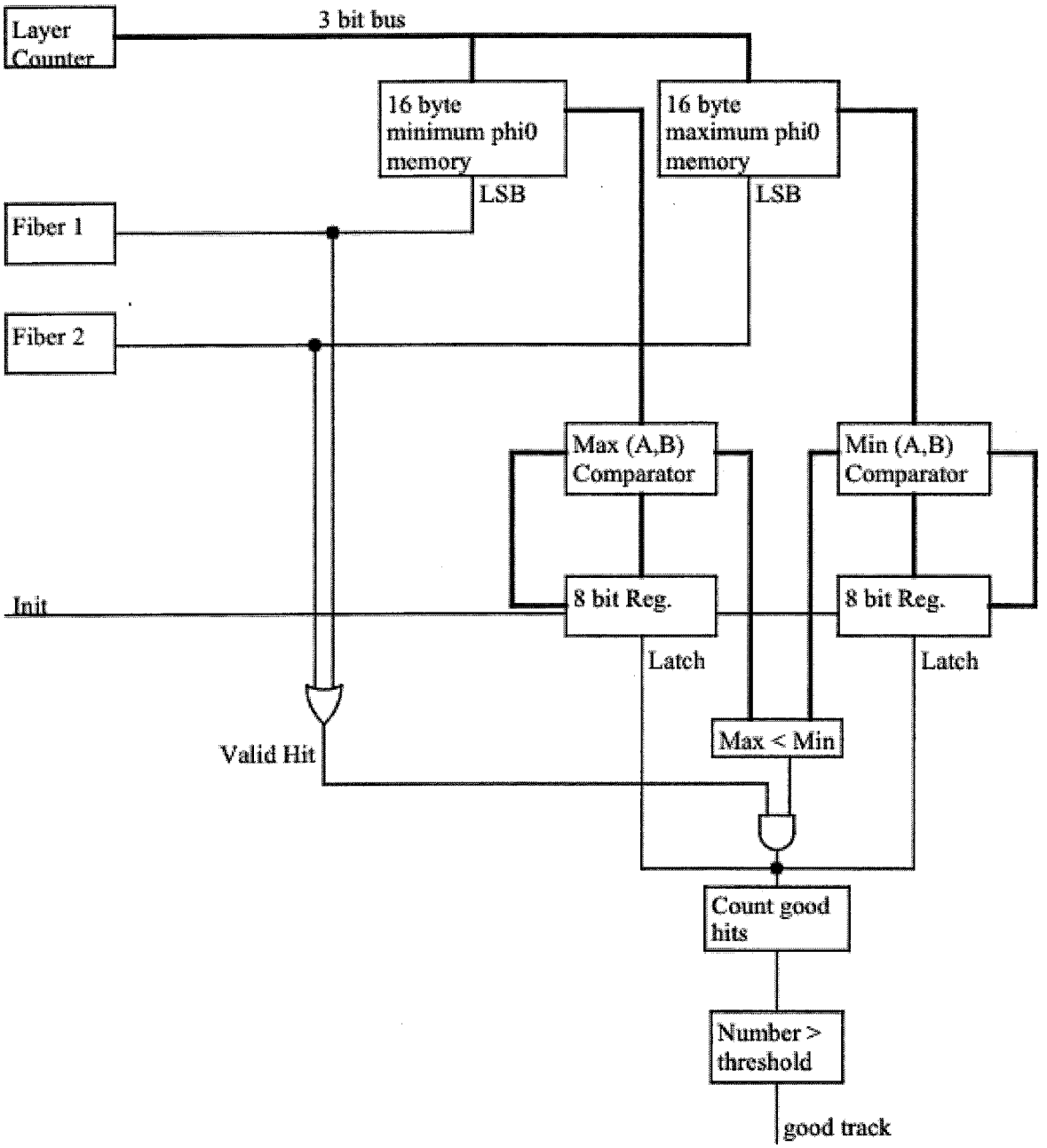

Fig. 4. Block diagram of the direct implementation of the algorithm for an 8 layer tracking detector. The memories store the ϕ_0's of the leading and trailing edges. The counter selects the layer. Not shown is the logic required to route the correct detector elements to this device.

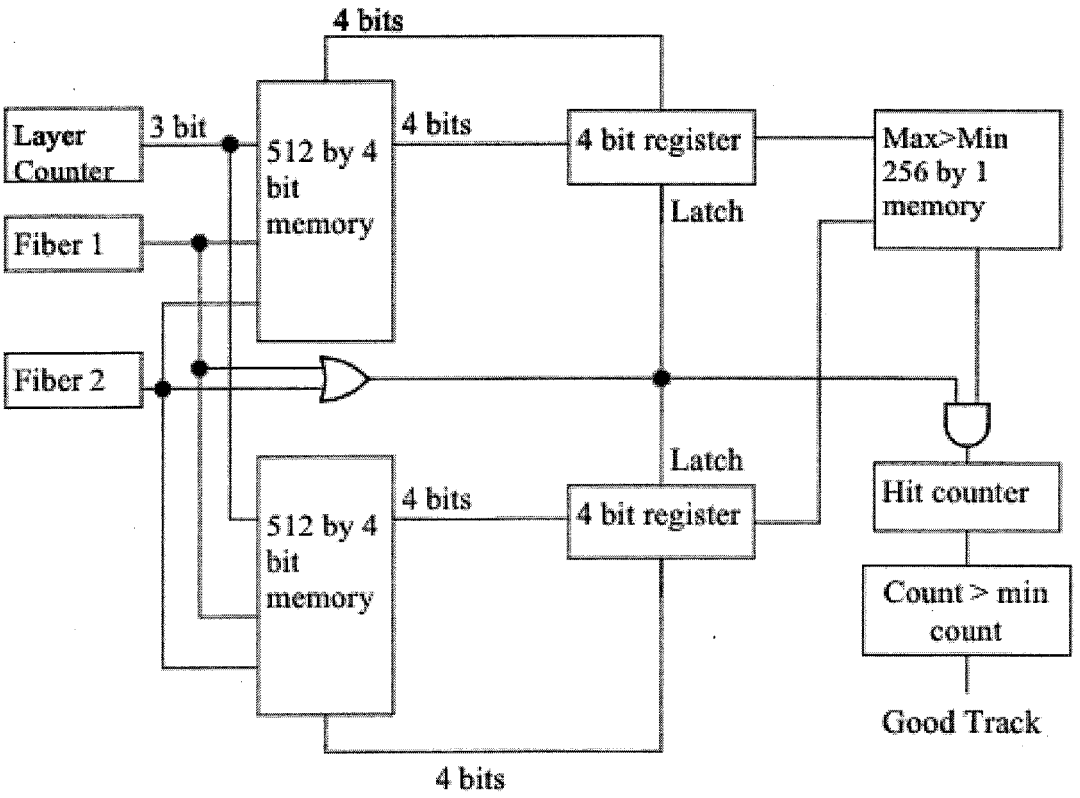

Fig. 5. Block diagram of a processor using only memory. This is again for an 8 layer detector. No requirement is placed on efficiency. The 2 memory units can be replaced with a single dual ported block provided that the layer counter is complemented on the second port. The 8 bit by 1 memory to determine if this is a good track could also be combined into the 512 by 4 memory by widening the memory to 5 bits and providing some additional logic to route the latch outputs into the address bits. Both fiber 1 and fiber 2 are shown as input bits. At the inner layer, there is only one fiber (the sorting is done on inner fiber outer fiber pair) so if one requires a hit in the first or second layer, then fiber 2 is not needed as an input bit; the absence of fiber 1 implies fiber 2. For the other layers if fiber 2 is absent, the output is not latched so the layer is skipped. This will reduce memory size to 256 by 4. Also note that even though there are 14 leading edge values and 14 trailing edge values, usually only a few dominate. Thus, searching though the completed table and removing redundant entries can further compress the table. A test sample for the 8 layer detector achieved more than a factor of 2 reduction. This would reduce the memory to 128 by 3.

TABLE I

Memory cell	0 No hit col.	1 L11 column	2 L21 column	3 L22 column	4 L31 column	5 L32 column	6 L41 column
Layer 1, Fiber 1	1 if present, 0 if absent	0	0	0	0	0	0
Layer 2, Fiber 1	2 if present, 0 if absent	Max[L21,L11] L21=2,L11=1	0	0	0	0	0
Layer 2, Fiber 2	3 if present, 0 if absent	Max[L22,L11] L22=3,L11=1	0	0	0	0	0
Layer 3, Fiber 1	4 if present, 0 if absent	Max[L31,L11] L31=4,L11=1	Max[L31,L21] L31=4,L21=2	Max[L31,L22] L31=4,L22=3	0	0	0
Layer 3, Fiber 2	5 if present, 0 if absent	Max[L32,L11] L32=5,L11=1	Max[L32,L21] L32=5,L21=2	Max[L32,L22] L32=5,L22=3	0	0	0
Layer 4, Fiber 1	6 if present, 0 if absent	Max[L41,L11] L41=6,L11=1	Max[L41,L21] L41=6,L21=2	Max[L41,L22] L41=6,L22=3	Max[L41,L31] L41=6,L31=4	Max[L41,L32] L41=6,L32=5	0
Layer 4,	7 if present,	Max[L42,L11]	Max[L42,L21]	Max[L42,L22]	Max[L42,L32]	Max[L42,L32]	0

Fiber 2	0 if absent	L42=7,L11=1	L42=7,L21=2	L42=7,L22=3	L42=7,L32=4	L42=7,L32=5	

Table 1. Memory lookup table for the leading edge for the first 4 layers. Each row is a layer and each column is a memory location. The notation L21 means layer 2, fiber 1. Max[L21,L11] means to take the maximum of layer 2, fiber 1 and layer 1, fiber 1. If L21 is the maximum, the contents of this cell is set to 2, otherwise it is set to 1. This number selects the column for the next layer. That is, if L21 were the maximum for layer 2, the output would be 2. At layer 3, it would compare L31 and L21 to select the maximum. This table makes no requirement on detector efficiency. There is a table like this for every road. The trailing edge table is similar and can be stored in the unused cells in this table. Note that this requires dual port memories.